Genetic Algorithm: The Eight Queens Problem

Bingling Fu

Final Draft: April 14, 2018

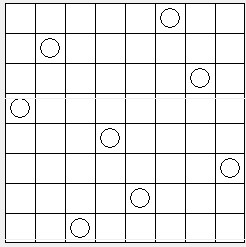
**Problem**

I created a genetic algorithm to find a solution to the Eight Queens Problem. The problem poses the situation: The Eight Queens question is a question on the background of chess: how can eight queens be placed on an 8×8 chess board so that no queen can directly eat other queens? In order to achieve this goal, neither of the two queens can be in the same horizontal, vertical or diagonal line.

**Design**

*Genetic code and expression:* With multi-value coding, the length of the chromosome depends on the number of queens. The position of each gene in the chromosome indicates the number of rows in the checkerboard, and the gene value indicates the number of columns.

It can be seen as a gene. This code can naturally solve the constraint that only one queen can be placed in a row. If every element is not repeated, it can be regarded as an arrangement of 0 n-1. It is natural to ensure that each column has only one queen. Therefore, it is necessary to pay attention to the uniqueness of list in cross mutation and generation of individuals. For example, the solution is 46031752, it means like that:

****

*Fitness function:* Since I am evaluating for the optimal solution, the fitness function is A cycle with a smaller weight will have a larger fitness value and vice versa.

*Mutation:* I am doing swap mutations of randomly selected bases. For example:

Parent: 1 | 7 | 2 | 4 | 5 | 3 | 6

swap indexes 1 and 2

Child: 7 | 1 | 2 | 4 | 5 | 3 | 6

I am also doing an insertion mutation, shifting three adjacent bases to a different location on the gene. This is done as:

Parent: 1 | 7 | 2 | 4 | 5 | 3 | 6

insert values starting at index 5 to index 2

Child: 1 | 5 | 3 | 6 | 7 | 2 | 4

Which version of mutation is performed on a child is selected at random.

*Crossing Over:* Since the individuals of the new population are randomly generated, the adjacent (and optionally randomly paired) individuals are chosen as parents to cross. Different from the bit-coded crossover, in order to ensure that the peers do not conflict with each other, the numbers in the new individual cannot be duplicated. First random generation of a single intersection position, the first half of the offspring individual is directly copied into the first half of the parent before the intersection, and the parent's parent is removed from the first half of the individual as the latter half of the individual.

*Mutation:* Randomly generate two gene positions, exchange genes at two positions and mutate individuals.

*Evolution:* The population is stored as a doubly linked list. The population is seeded with 4 individuals with genes that have been shuffled with the Fisher-Yates method. For each generation, all individuals sexually reproduce, doubling the population, and each child also experiences one of the two mutation methods. I then calculate the fitness function for each individual and write those values to an array. I perform an insertion sort on the array (selected since the graph sizes being evaluated are relatively small), randomly generate a culling ratio between 10% and 90% and remove that proportion of the weakest individuals from the population. The remaining individuals then reproduce, doubling the population again.

For each generation, after the culling is complete the number of survivors is logged along with the highest fitness score. The evolutionary process terminates after 100 generations or when the highest fitness score hasn't changed in 10 generations.